Amendments to the Specification:

On Page 1, lines 9-12, please amend as follows:

This applications claims priority Under U.S.C. 119(e) from U.S. Provisional Patent Application No. 60/235,190 entitled "E-Commerce Security Processor" filed on ~~September 20, 2000~~ September 25, 2000 which is incorporated by reference in its entirety for all purposes.

On page 8. lines 4-22, please amend as follows:

During operation, the inventive encryption accelerator 302 implements the ARCFOUR algorithm by requiring that a state memory 316 be initialized with an incrementing pattern (i.e., location 0 contains the value 0, location 1 contains the value 1, and so on). In the described embodiment, the state memory 316 is 256 bytes in size. In a shuffling operation, a secret key array 318 that is stored in the system memory [[310]] 312 is used to move state memory values to new locations in the state memory 316. In the described embodiment, the secret key array 318 consists of 256 bytes, where each byte is 8 bits. The secret key array 318 is produced by repeating the secret key until 256 bytes are filled. In this way, the values in the state memory 316 at the end of the shuffling operation consist of the numbers 0 through 255, but the locations of those values in the state memory 316 are only known if the secret key array

318 is known. In this way, this inventive accelerator 302 produces the initial incrementing state memory pattern totally in hardware whereas the shuffling operation is performed by transferring the secret key array 318 and an associated message data length (in bytes) into the accelerator 302 via the system bus 308 and any intervening external interfaces thereby preserving valuable CPU resources. It should be noted that the shuffling operation in the state memory 316 is performed "on the fly" as transfer of the secret key array 318 takes place.

In paragraph beginning on page 8, line 30, and ending on page 9 line 11, please amend as follows:

As noted above, the encryption accelerator 302 is capable of operating in multiple modes that include an Initial Mode and a Continuation Mode. When the accelerator is operation in the Initial Mode, the operations described above are performed sequentially. However, as shown in Fig. 4, when in the Continuation mode, the state memory 316 is reloaded with the contents of the state memory 316 that were saved to external memory (such as the system memory [[310]] 312, if so desired) when a Last Transfer flag is not set when an earlier stream of data was interrupted. For example, when the accelerator 302 is processing a first data stream that is interrupted at t=t0, the contents of the state memory 316 as it stood at t=t0 are stored externally (if the Last Transfer flag is not set) and processing of a second data stream is then commenced at approximately t=t1. At the completion of the processing of the second data stream

t=t2, the contents of the state memory 316 as it stood at t=t0 corresponding to state of processing of the interrupted first data stream at t=t0 is restored to the state memory 316. At this point, the processing of the first data stream can be restarted at approximately t=t3.

On page 9, lines 26-30, please amend as follows:

During operation, the state machine 502 directs the shuffling operation in the state memory 316 by causing the secret key array 318 to be retrieved from the system memory [[310]] 312 and directing the counters 506 and 508 to increment the indices (i,j) accordingly. In this way, the shuffling operations are completely performed by the accelerator 302 thereby preserving valuable CPU resources.

On page 11, lines 2-12, please amend as follows:

Fig. 7 shows a flowchart detailing a process 700 for implementing the ciphering operation 618 of the process 600 shown in Fig. 6. The process [[800]] 700 begins at 702 by receiving a byte of the data to be encrypted and at 704 by incrementing the index variable i by one. Next, at 706, the contents of the i^{th} element of the state memory is added to the j^{th} element of the state memory while at 708 the i^{th} and j^{th} elements of the state memory are swapped. At 709, the i^{th} and j^{th} elements of the state memory are added together to form a new value n. At 710, an encrypted output byte is formed by

combining the nth element of the state memory with the data byte by encrypted using a bit by bit exclusive OR operation. At 712, a determination is made whether or not there are additional bytes to be encrypted. If there are additional bytes, then control is passed back to 702, otherwise processing is stopped.